VAX/VMS Bad Block Locator Utility Reference Manual

Order No. AA-Z435A-TE

digital software

VAX/VMS
Bad Block Locator Utility
Reference Manual

Order Number: AA–Z435A–TE

September 1984

This document describes the Bad Block Locator Utility for use on VAX processors.

Revision/Update Information: This is a new manual.

Software Version: VAX/VMS Version 4.0

digital equipment corporation
maynard, massachusetts

September 1984

The postpaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC	DIBOL	UNIBUS
DEC/CMS	EduSystem	VAX
DEC/MMS	IAS	VAXcluster
DECnet	MASSBUS	VMS
DECsystem–10	PDP	VT
DECSYSTEM–20	PDT	
DECUS	RSTS	
DECwriter	RSX	digital

ZK–2305

This document was prepared using an in-house documentation production system. All page composition and make-up was performed by TEX, the typesetting system developed by Donald E. Knuth at Stanford University. TEX is a registered trademark of the American Mathematical Society.

BAD Contents

INDEX

TABLES

Preface

Intended Audience

This manual is intended for VAX/VMS system managers, operators, and system programmers.

Structure of This Document

This document is composed of three major sections.

The Format Section is an overview of BAD and is intended as a quick reference guide. The format summary contains the DCL command that invokes BAD, listing all qualifiers and keywords. The usage summary describes how to invoke and exit from BAD, how to direct output, and any restrictions you should be aware of.

The Description Section explains how to use BAD.

The Qualifier Section describes each DCL command qualifier. Qualifiers appear in alphabetical order.

Associated Documents

For additional information on the topics covered in this document, refer to the *VAX/VMS DCL Dictionary* and the *Guide to VAX/VMS System Management and Daily Operations*.

Conventions Used in This Document

Convention	Meaning
RET	A symbol with a one- to three-character abbreviation indicates that you press a key on the terminal, for example, RET .
CTRL/x	The phrase CTRL/x indicates that you must press the key labeled CTRL while you simultaneously press another key, for example, CTRL/C, CTRL/Y, CTRL/O.
$ SHOW TIME 05-JUN-1985 11:55:22	Command examples show all output lines or prompting characters that the system prints or displays in black letters. All user-entered commands are shown in red letters.

Convention	Meaning
$ TYPE MYFILE.DAT . . .	Vertical series of periods, or ellipsis, mean either that not all the data that the system would display in response to the particular command is shown or that not all the data a user would enter is shown.
file-spec,...	Horizontal ellipsis indicates that additional parameters, values, or information can be entered.
[logical-name]	Square brackets indicate that the enclosed item is optional. (Square brackets are not, however, optional in the syntax of a directory name in a file specification or in the syntax of a substring specification in an assignment statement.)
quotation marks apostrophes	The term quotation marks is used to refer to double quotation marks ("). The term apostrophe (') is used to refer to a single quotation mark.

New and Changed Features

The Bad Block Locator Utility no longer has an interactive interface. BAD functions are performed using the DCL command ANALYZE/MEDIA, along with the qualifiers documented here.

BAD

The Bad Block Locator Utility (BAD) analyzes block-addressable devices and records the locations of blocks that cannot reliably store data. Table BAD–1 at the end of the description section describes the devices that BAD can analyze.

FORMAT

ANALYZE/MEDIA *device*

Command Qualifiers	Defaults
/BAD_BLOCKS[=(list)]	*None.*
/[NO]EXERCISE[=(keyword[,...])]	*/NOEXERCISE*
/[NO]LOG	*/NOLOG*
/OUTPUT[=file-spec]	*None.*
/[NO]RETRY	*/NORETRY*
/SHOW[=(keyword[,...])]	*None.*

Command Parameter

device

Specifies the device containing the volume that BAD will analyze. The device name has the form

```
ddcu: or logical-name
```

usage summary

Invoking

To invoke BAD, at the DCL command prompt enter the command ANALYZE /MEDIA along with any parameters or qualifiers.

Exiting

Once invoked, BAD will run until completion. When BAD terminates, control is returned to the DCL command level.

Directing Output

You can write the contents of the Detected Bad Block File (DBBF) to an output file by specifying the /OUTPUT qualifier. See the description of the /OUTPUT qualifier in the qualifier section.

Privileges/Restrictions

To ensure that the device is not accessed by any other programs, you must allocate the device with the DCL command ALLOCATE. See the *VAX/VMS DCL Dictionary* for more information on the ALLOCATE command.

After you have allocated the device, you must give the DCL command MOUNT with the /FOREIGN qualifier. When the device is mounted foreign, the operating system does not recognize it as a Files–11 volume, and BAD can execute.

DESCRIPTION

BAD locates bad blocks on a volume by testing whether the same data that is written onto blocks can be read back. When it finds a bad block, BAD writes the address of that block onto the Software Detected Bad Block File (SDBBF).

When you run BAD to test a device (using the /EXERCISE qualifier), keep in mind that:

- The device cannot be accessed by other programs,
- The device cannot be mounted as a Files–11 volume,
- The device is always purged by BAD's testing procedure; any information stored on the disk is destroyed.

Caution: There is no way to test the volume for bad blocks without destroying its contents. However, you can update the DBBF without erasing the volume's contents by using the BAD qualifiers /NOEXERCISE and /BAD_BLOCKS.

After BAD locates and records the bad blocks, you issue the DCL command INITIALIZE to change the volume from unstructured format to Files–11 format and to allocate the faulty blocks to a special file on the volume called [000000]BADBLK.SYS. In this way, users are protected from accessing them for their files. For more information on Files–11 format and the INITIALIZE command, see the *VAX/VMS DCL Dictionary*.

1 Locating Bad Blocks

To test the blocks on a volume, BAD

- Writes a test pattern to each block on the disk
- Reads the contents of the block into a buffer
- Compares the data read back with the data written

If the data does not compare exactly, a block cannot reliably store data.

2 Recording Bad Blocks

When BAD locates a bad block, it records the address of the block. Consecutive bad blocks are recorded as single entries for non-last-track devices. After it finishes testing the disk, BAD writes the addresses of the bad blocks into an area called the Detected Bad Block File (DBBF).

2.1 Location of the Detected Bad Block File

The location of the DBBF depends on whether the volume is a last-track device, that is, one that contains more than 4096 blocks (512 bytes per block). Last-track devices store bad block data on the last track of the disk.

The first half of the track is reserved for the Manufacturer's Detected Bad Block File (MDBBF). The MDBBF stores the bad blocks discovered by the manufacturer when the device was originally formatted.

The second half of the track is reserved for the Software Detected Bad Block File (SDBBF).

Non-last-track devices are devices that contain 4096 blocks (512 bytes per block) or less. These devices do not set aside the last track of the disk to store bad block information. Instead, BAD creates the DBBF on the last good block of the disk. There must be at least one reliable block in the last 256 blocks of the volume for BAD to generate the DBBF.

2.2 Format of the Detected Bad Block File

If the volume is a last-track device, each DBBF entry contains the cylinder, track, and sector address of the faulty block.

On volumes that are not last track, DBBF entries contain the number of bad blocks minus 1, and the logical block number (LBN) of the faulty block or sequence of faulty blocks. A single entry can address one bad block or several contiguous bad blocks.

The DBBF can contain a maximum of 126 entries. For both last-track and non-last-track devices, BAD terminates with an error message when the maximum number of entries is exceeded. However, the contents of the SDBBF on the medium remains intact.

3 Running BAD on Devices Converted to Non-Last-Track Format

If you have run the compatibility mode BAD using the /OVR (override) qualifier on a device designated as last-track, the medium on the device has been converted to non-last-track format. When you run the native mode BAD on such a device, BAD treats it as a last-track device and will therefore fail to find the Manufacturer's Detected Bad Sector File (MDBSF). Depending on which qualifiers you specify with the ANALYZE/MEDIA command, you can expect the following results when running BAD on a converted device:

- If you do not attempt to modify the medium on the device (if you specify /NOEXERCISE /OUTPUT, for example), BAD will, upon failing to locate the MDBSF, attempt to find the Software Detected Bad Sector File (SDBSF) and will produce an output listing using the non-last-track format.
- If you attempt to modify the medium preserving the SDBSF (if you specify /EXERCISE=KEEP or /NOEXERCISE /BAD_BLOCKS, for example), BAD will terminate with the following fatal diagnostic message:

```
% BAD-F-MDBSFRFAIL, Failed to read the manufacturer's detected bad sector file on DEVICENAME
```

Note that all attempts to preserve the SDBSF when modifying the medium on a converted device will fail. However, if you do not attempt to preserve the SDBSF (by specifying /EXERCISE=KEEP, for example), BAD will recreate the MDBSF and SDBSF and will continue testing. For example, you could use the following command to run BAD on the converted device DB1:

```
$ ANALYZE/MEDIA/OUTPUT=DB1TEST.ANL DB1:
ANALYZE/MEDIA                          15-APR-1984 08:16:53.97   PAGE 1
DB1: (DBA1:), SERIAL NUMBER: 0

SOFTWARE DETECTED BAD BLOCKS

LOGICAL BLOCK NUMBER    COUNT
   88308                  0

DEVICE DBA1:CONTAINS A TOTAL OF 340670 BLOCKS; 1 DEFECTIVE BLOCK DETECTED
ANALYZE/MEDIA/EXERCISE/OUTPUT=DB1TEST.ANL DB1:
```

4 Running BAD Interactively from Your Terminal

When you run BAD interactively from your terminal, use a sequence of commands similar to the following:

```
$ ALLOCATE DBA2:
DBA2: ALLOCATED
$ MOUNT/FOREIGN DBA2:
%MOUNT-I-MOUNTED     mounted on DBA2:
$ ANALYZE/MEDIA/EXERCISE DBA2:
$ INITIALIZE DBA2:
```

The ALLOCATE command requests the allocation of a specific disk drive, DBA2:. The response from the ALLOCATE command indicates that the device was successfully allocated. The MOUNT/FOREIGN command mounts the disk volume as a foreign disk. The MOUNT command response indicates that DBA2: was successfully mounted. The ANALYZE/MEDIA command invokes BAD. Specifying DBA2: and the /EXERCISE qualifier causes BAD to analyze each block on this disk volume and to record the bad blocks. The INITIALIZE command reformats the volume as a Files–11 volume and allocates the bad blocks to a special file on the volume called [000000]BADBLK.SYS. Once allocated, they cannot be used by other files.

5 Running BAD from Command Procedures

You can invoke BAD from a VAX/VMS command procedure. Following is a typical command procedure, named NEWDISK.COM, that invokes BAD and gives other DCL commands.

```
$! THIS COMMAND PROCEDURE EXERCISES AND INITIALIZES THE
$! SPECIFIED DISK.
$!
$! P1 IS THE DEVICE
$! P2 IS THE VOLUME LABEL
$!
$!
$ IF P1 .EQS. "" THEN $ INQUIRE P1 "DEVICE TO BE INITIALIZED"
$ IF P2 .EQS. "" THEN $ INQUIRE P2 "VOLUME LABEL"
$ ALLOCATE 'P1
$ MOUNT/FOREIGN 'P1
$ ANALYZE/MEDIA/EXERCISE 'P1
$ DISMOUNT/NOUNLOAD 'P1
$ INITIALIZE 'P1 'P2
$ EXIT
```

To run this command procedure, type the following command in response to the DCL prompt:

```
@NEWDISK device volume-label
```

The operating system executes the commands in the order given within the command procedure.

6 Error Messages

The *VAX/VMS Utilities Reference Volume* lists the diagnostic messages generated by BAD, and provides explanations and suggested user actions for these messages.

Table BAD–1 Block-Addressable Devices

Model	Code	Type	❶ RPM	Surfaces	Cylinders	Bytes/Track	Bytes/Drive
RB02	DQ	Cart	2400	2	512	10,240	10,485,760
RL02	DL	Cart	2400	2	512	10,240	10,485,760
RM03	DR	Pack	3600	5	823	16,384	67,420,160
RM05	DR	Pack	3600	19	823	16,384	256,196,608
RB80	DQ	Fix	3600	14	559	15,872	124,214,272
RM80	DR	Fix	3600	14	559	15,872	124,214,272
RP05	DB	Pack	3600	19	411	11,264	87,960,576
RP06	DB	Pack	3600	19	815	11,264	174,423,040
RP07	DR	Fix	3600	16 ❷	630	25,600	516,096,000
RK06	DM	Cart	2400	3	411	11,264	13,888,512
RK07	DM	Cart	2400	3	815	11,264	27,540,480
RX01	DX	Flop	360	1	77	3,328	256,256
RX02	DY	Flop	360	1	77	3,328 ❸ 6,656 ❹	256,256 ❸ 512,512 ❹
TU58 ❺	DD	Cart	---	---	---	---	262,144

❶ Pack = pack disk; Cart = cartridge disk; Flop = floppy (flexible diskette); Fix = fixed media.

❷ The RP07 has 16 surfaces but 32 tracks per cylinder.

❸ Single density.

❹ Double density.

❺ A magnetic tape device, the TU58 operationally resembles a disk device.

COMMAND QUALIFIERS

This section presents qualfiers for the ANALYZE/MEDIA command in alphabetical order. The qualifiers follow the standard rules of DCL grammar, as specified in the *VAX/VMS DCL Dictionary*. Thus, you can abbreviate any qualifier or keyword as long as the abbreviation is not ambiguous. The asterisk and the percent sign can be used as wildcard characters unless otherwise noted.

/BAD_BLOCKS

Adds the specified bad blocks to the DBBF. If the /BAD_BLOCKS qualifier is specified along with the /EXERCISE qualifier, the medium is tested once the bad blocks are added.

FORMAT /BAD_BLOCKS*[(=list)]*

qualifier keyword

list

Specifies codes for the the bad block locations to be added to the DBBF. If you do not specify a value for the /BAD_BLOCKS qualifier, BAD prompts as follows:

```
BAD_BLOCKS =
```

When it prompts, BAD reports any duplicate bad blocks. To terminate the prompting session, type CTRL/Z.

Note: The term "block" denotes a standard unit of 512 bytes, whereas the term "sector" denotes the physical size of the device sector, which is not always the same for all devices. For example, an RL02 has a sector size of 256 bytes, while an RK07 has a standard sector size of 512 bytes.

Valid bad block location codes follow. You can specify them in any integer combination or radix combination.

Code	Meaning
lbn	Specifies the logical block number (LBN) of a single bad block.
lbn:count	Specifies a range of contiguous bad blocks starting at the logical block number (LBN) and continuing for "count" blocks.
sec.trk.cyl	Specifies the physical disk address (sector, track, and cylinder) of a single bad sector. This code is valid only for last track devices.
sec.trk.cyl:count	Specifies a range of bad sectors starting at the specified physical disk address (sector, track, and cylinder) and continuing for "count" sectors. This code is valid only for last track devices.

EXAMPLES

1
```
$ ANALYZE/MEDIA/BAD_BLOCKS=(4.4.4:3) DB1:
```
The /BAD_BLOCKS qualifier in this example specifies a range of 3 bad sectors beginning at the physical disk address sector 4, track 4, cylinder 4. This range is added to the DBBF.

2
```
$ ANALYZE/MEDIA/EXERCISE/BAD_BLOCKS=(2) DB1:
```
The command in this example adds the bad block specification to the DBBF and then tests the medium. The bad block in this example is located at logical block number (LBN) 2.

```
3   $ ANALYZE/MEDIA/EXERCISE/BAD_BLOCKS DB1
    BAD_BLOCKS = 2 3
    BAD_BLOCKS = 4
    % BAD-I-DUPBLKNUM, duplicate block number 4, already exists in SDBBF
    -BAD-I-SRCLIN, the source input entry was 4
    BAD_BLOCKS =
        .
        .
        .
    BAD_BLOCKS= CTRL/Z
```

In this example, BAD prompts for bad block specifications. Note that, in prompt mode, BAD reports any duplicate bad blocks it detects.

/EXERCISE

Controls whether the medium should actually be tested. The default is /NOEXERCISE.

FORMAT

/EXERCISE *[=(keyword[,...])]*
/NOEXERCISE

qualifier keywords

FULL

Causes BAD to test the medium using three test patterns (0's, 1's, and "worst case"), instead of the default single "worst case" pattern. The FULL keyword can be used only with /EXERCISE. Note that the "worst case" pattern always remains on media tested with the /EXERCISE qualifier.

KEEP

Ensures the preservation of the current SDBBF. The KEEP keyword is the default when /NOEXERCISE is specified.

NOKEEP

Causes BAD to create a new SDBBF. The NOKEEP keyword is the default when /EXERCISE is specified. This keyword cannot be used with the /NOEXERCISE qualifier.

PATTERN=(longword[,...])

Allows users to specify the value of a test pattern to be used as "worst case." Up to an octaword of test pattern data may be specified in decimal (%D), hexadecimal (%X), or octal (%O) radixes. The default radix is decimal.

The pattern is specified in longwords. If two or more longwords are specified, they must be enclosed in parentheses and separated by commas.

EXAMPLES

1 `$ ANALYZE/MEDIA/EXERCISE=FULL DB1:`

The command in this example tests the medium using three test patterns. By default, a new SDBBF is created.

2 `$ ANALYZE/MEDIA/EXERCISE=KEEP DB1:`

This command tests the medium while preserving the current SDBBF.

3 `$ ANALYZE/MEDIA/EXERCISE=PATTERN=(%XFFEEFFEE,%XBADBADBA) DB1:`

This command specifies a hexadecimal test pattern two longwords in length.

4 `$ ANALYZE/MEDIA/NOEXERCISE/BAD_BLOCKS DB1:`

This command updates the DBBF without erasing the volume's contents.

/LOG

Specifies whether a message is sent to SYS$OUTPUT and to SYS$ERROR indicating the total number of bad blocks detected by BAD. The default is /NOLOG.

FORMAT /[NO]LOG

EXAMPLE

```
$ ANALYZE/MEDIA /LOG DBB1:
Device DBB1: contains a total of 340670 blocks; 11 defective blocks detected.
```

The command in this example requests BAD to report the total number of bad blocks it detected on the device DBB1:.

/OUTPUT

Specifies whether the contents of the DBBF are written to the specified file. If you omit the /OUTPUT qualifier, no output is generated.

When you specify /OUTPUT in conjunction with the /SHOW qualifier, the default keyword for the /SHOW qualifier is AFTER.

FORMAT

/OUTPUT *=[file-spec]*

qualifier keyword

file-spec

Identifies the output file for storing the results of the medium analysis. If you specify a file type and omit the file name, the default file name ANALYZE is used. The default file type is ANL. If you omit the file-spec, the results are output to SYS$OUTPUT.

In place of the file-spec, you may specify an output device. In this case, BAD writes the contents of the volume's DBBF to a file called ANALYZE.ANL and queues the file to the output device.

No wildcard characters are allowed in the file specification.

EXAMPLES

1
```
$ ANALYZE/MEDIA/OUTPUT=BADDBB.DAT DBA2:
```
The command in this example writes the contents of the DBBF from DBA2: to the output file BADDBBF.DAT. Note that since /NOEXERCISE is the default, the medium is not tested.

2
```
$ ANALYZE/MEDIA/OUTPUT=LPB0: DBA2:
```
The command in this example writes the contents of the DBBF from DBA2: to the default file ANALYZE.ANL and queues the file to the print device LPB0:.

/RETRY

Enables the device driver to retry soft errors. The /RETRY qualifier is used only in conjunction with the /EXERCISE qualifier. The default is /NORETRY.

FORMAT

/EXERCISE /NORETRY
/EXERCISE /RETRY

EXAMPLE

```
$ ANALYZE/MEDIA /EXERCISE /RETRY DBA0:
```

The command in this example directs the device driver to retry soft errors.

/SHOW

Lists the contents of the DBBF before and/or after the medium is exercised (tested).

FORMAT

/SHOW *[=(keyword[,...])]*

qualifier keywords

[NO]BEFORE,[NO]AFTER

Specifies whether the contents of the DBBF is listed before and/or after the medium is exercised (tested). AFTER is the default.

EXAMPLES

1 `$ ANALYZE/MEDIA/EXERCISE/OUTPUT/SHOW=(BEFORE,AFTER) DBA3:`

The command in this example lists the contents of the DBBF both before and after the disk DBA3: is exercised and directs the data to the currect SYS$OUTPUT device.

2 `$ ANALYZE/MEDIA/EXERCISE/OUTPUT/SHOW DBA3:`

The command in this example lists the contents of the DBBF only after the disk DBA3: is exercised and directs the data to SYS$OUTPUT.

Index

B

D

E

I

L

O

R

S

VAX/VMS Bad
Block Locator Utility
Reference Manual
AA-Z435A-TE

READER'S COMMENTS

Note: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

_____

_____

_____

_____

_____

_____

_____

_____

_____

Did you find errors in this manual? If so, specify the error and the page number.

_____

_____

_____

_____

_____

_____

_____

_____

_____

Please indicate the type of user/reader that you most nearly represent:

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) _____

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code _____
or Country

Do Not Tear - Fold Here and Tape

digital

No Postage Necessary if Mailed in the United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

SSG PUBLICATIONS ZK1-3/J35
DIGITAL EQUIPMENT CORPORATION
110 SPIT BROOK ROAD
NASHUA, NEW HAMPSHIRE 03062-2698

Do Not Tear - Fold Here

Cut Along Dotted Line